Customized Sequential Designs for Random Simulation Experiments

van Beers, W.C.M.; Kleijnen, J.P.C.

Publication date:
2005

No. 2005–55

CUSTOMIZED SEQUENTIAL DESIGNS FOR RANDOM SIMULATION EXPERIMENTS: KRIGING METAMODELING AND BOOTSTRAPPING

By Wim C.M. van Beers, Jack P.C. Kleijnen

March 2005

ISSN 0924-7815

Customized Sequential Designs for Random Simulation Experiments: Kriging Metamodeling and Bootstrapping

Wim C.M. van Beers
Department of Information Systems and Management
Tilburg University (UvT), Postbox 90153, 5000 LE Tilburg, The Netherlands
Phone: +31-13-4668202; Fax: +31-13-4663069; E-mail: wvbeers@uvt.nl

Jack P.C. Kleijnen
Department of Information Systems and Management/
Center for Economic Research (CentER)
Tilburg University (UvT), Postbox 90153, 5000 LE Tilburg, The Netherlands
Phone: +31-13-4662029; Fax: +31-13-4663069; E-mail: kleijnen@uvt.nl
http://center.uvt.nl/staff/kleijnen/

This paper proposes a novel method to select an experimental design for interpolation in random simulation, especially discrete event simulation. (Though the paper focuses on Kriging, this design approach may also apply to other types of metamodels such as linear regression models.) Assuming that simulation requires much computer time, it is important to select a design with a small number of observations (or simulation runs). The proposed method is therefore sequential. Its novelty is that it accounts for the specific input/output behavior (or response function) of the particular simulation at hand; i.e., the method is customized or application-driven. A tool for this customization is bootstrapping, which enables the estimation of the variances of predictions for inputs not yet simulated. The new method is tested through two classic simulation models: example 1 estimates the expected steady-state waiting time of the M/M/1 queueing model; example 2 estimates the mean costs of a terminating (*s*, *S*) inventory simulation. For these simulations the novel design indeed gives better results than Latin Hypercube Sampling (LHS) with a prefixed sample of the same size.

Key words: Simulation: design of experiments, statistical analysis, Kriging, bootstrapping, regression, C0, C1, C9, C15, C44.

1. Introduction

In this paper, we focus on *expensive simulations*; that is, we assume that a single simulation run takes 'much' computer time. Consequently, 'interpolation' is needed; i.e., from the simulated input/output (I/O) data, the outputs are predicted for input combinations not yet simulated. We devise a method that is meant to minimize the number of simulation runs for such interpolation. We *tailor* our design of experiments (DOE) to the actual simulation; that is, we do not derive a generic design such as a classic design (for example, a 2^{k-p} design) or a LHS design. The differences between customized and generic designs are as follows (also see Kleijnen and Van Beers (2004), who focus on deterministic simulation).

A *metamodel* is a model of the I/O function (or 'response function') implied by the underlying simulation model. We denote the metamodel by $Y(\mathbf{x})$ where $\mathbf{x}$ denotes the *k*-dimensional vector of the *k* inputs (factors) so $\mathbf{x} = (x_1, \ldots, x_j, \ldots, x_k)'$. *Classic DOE* assumes a simple metamodel. For example, designs of resolution III (including certain 2^{k-p} designs) assume a first-order polynomial I/O function. Composite designs (CCD) assume a second-order polynomial. These designs are discussed for physical experiments in (for example) the well-known textbook Box, Hunter, and Hunter (1978) and the recent textbook Myers and Montgomery (2002); for simulation experiments we refer to Kleijnen (1987).

*LHS (*much applied in Kriging, described below) assumes that an adequate metamodel is more complicated than a low-order polynomial. LHS, however, does not assume a specific metamodel. Instead, LHS focuses on the design space formed by the *k*–dimensional unit cube, defined by $0 \le x_j \le 1$ ($j = 1, \ldots, k$) after standardizing (scaling) the inputs. LHS is one of the *space filling* designs: LHS samples that space according to some prior distribution for the inputs, such as independent uniform distributions on [0, 1]; see McKay, Beckman, and Conover (1979), and also Kleijnen et al. (2005), Koehler and Owen (1996), and Santner, Williams, and Notz (2003).

Unlike LHS, we explicitly account for the I/O function. Unlike classic DOE, we assume that a low-order polynomial (estimated through regression analysis) gives an inadequate approximation of the I/O function. In our method we estimate the uncertainty of predicted outputs at unobserved input combinations (these combinations are also called scenarios, design points, combinations of factor levels, or simulation inputs). To estimate the uncertainty of these predictions—caused by the noise and the shape of the I/O function—we use *bootstrapping*; i.e., we resample the outputs for each scenario already simulated (for

bootstrapping in general see the classic textbook, Efron and Tibshirani 1993; for bootstrapping in the validation of regression metamodels in simulation see Kleijnen and Deflandre 2005).

We make our procedure *sequential* for the following two reasons.

1. Sequential statistical procedures are known to be more 'efficient'; that is, they require fewer observations than fixed-sample (one-shot) procedures; see, for example, the handbook by Ghosh and Sen (1991) and the recent article by Park et al. (2002).
2. Simulation experiments proceed sequentially (unless parallel computers are used; our procedure also fits parallel computers).

The literature on *deterministic* simulation shows several designs that—like ours—account for the specific simulation's I/O function, and are sequential. For example, Crary (2002) discusses G-optimal and I-optimal designs, which the DOE literature defines as follows. G-optimal designs minimize the *maximum* Mean Squared Error (MSE) of the predicted output; I-optimal or Integrated MSE (IMSE) designs minimize the *average* MSE (obviously, the MSE reduces to the variance if the predictor is unbiased; see (5) and (6) below). Williams, Santner, and Notz (2000, 2002) use a Bayesian approach to derive sequential IMSE designs. Sasena, Papalambros, and Govaerts (2002) derive sequential designs for the optimisation of deterministic simulation models. Kleijnen and Van Beers (2004) derive customized sequential designs for deterministic simulations. We, however, focus on DOE for random simulations, and we seem to be the first to apply bootstrapping for this problem. (Random simulation includes Discrete Event Dynamic Systems or DEDS simulation such as M/M/1 simulation, but also simulation models consisting of stochastic difference equations.)

We shall see that our designs select most of their input combinations in sub-areas that have *more interesting* I/O behavior. In our first example we spend most of our computer simulation time on the challenging 'explosive' part of the metamodel that estimates the mean steady-state waiting time for various traffic rates of single-server queueing systems with Markovian (Poisson) arrival and service times—known as the M/M/1 model. (The reader may take a peek at Figure 1, discussed in subsection 5.1.) In our second example, we estimate the average total costs in an (*s*, *S*) inventory model; there are several variations on this model, but we take the specification given by Law and Kelton (2000). Again, we find a concentration of the input combinations in the sub-area where the metamodel shows steep slopes. (See Figure 7, detailed in subsection 5.2.) In both examples, we compare our designs with LHS; our designs give better predictions.

The remainder of this paper is organized as follows. Section 2 summarizes the basics of Kriging. Section 3 summarizes DOE and Kriging. Using the M/M/1 model, section 4 explains our method, which applies bootstrapping—to estimate the variances of the Kriging predictions for candidate inputs not yet simulated—and sequentially selects as the next input to be simulated, the one with the largest bootstrap variance. Section 5 demonstrates the procedure through two classic examples: subsection 5.1 uses M/M/1 simulations, and subsection 5.2 uses an (*s*, *S*) inventory model with two inputs. For both examples our method gives better results than LHS with a prefixed sample size. Section 6 presents conclusions and topics for further research.

2. Kriging basics

Kriging (named after the South-African mining engineer Krige) is an interpolation method that predicts unknown values of a random function or random process; see Journel and Huijbregts (1978) and Cressie's (1993) classic Kriging textbook on spatial (geo)statistics. Whereas spatial statistics considers the two-dimensional 'location' as the known input of this process, simulation considers the *k*–dimensional 'scenario' as input; see Sacks et al.'s (1989) classic article on the Design and Analysis of Computer Experiments (DACE)—these computer experiments concern deterministic simulation. Random (stochastic) simulation—including DEDS simulations—is the topic of our paper.

More precisely, a Kriging prediction is a weighted linear combination of all output values already observed. The weights depend on the distances between the new input to be predicted and the old inputs already observed. Kriging assumes that *the closer the inputs are, the more positively correlated the outputs are*. Mathematical formulations follow in equations (1) through (4).

Currently, Kriging is frequently applied in *deterministic simulation*, which is much used in engineering; again see Sacks et al. (1989); for an update see Simpson et al. (2001). In deterministic simulation, Kriging has an important advantage over regression analysis: the predicted values at old inputs are exactly equal to the observed (simulated) outputs.

In *random simulation*, however, this property disappears. Now, each scenario is simulated several times—with non-overlapping pseudo-random number (PRN) streams. Van Beers and Kleijnen (2003) show that Kriging interpolates the *average* output per scenario. These averages, however, are still random, so the property that at scenarios already simulated the Kriging predictions equal the averages, loses its intuitive appeal. Still, Kriging may be

attractive because it may decrease the prediction *bias* (and hence the MSE) at scenarios close together. Indeed, in the examples presented by Van Beers and Kleijnen (2003) the Kriging predictions are much better than the regression predictions (regression analysis may be useful for other goals such as screening and validation; see Kleijnen et al. 2004). Therefore we do not further discuss regression analysis in this paper.

Mathematically formulated, Kriging assumes the following metamodel:

$$Y(\mathbf{x}) = \mu(\mathbf{x}) + \delta(\mathbf{x}) \text{ with } \delta(\mathbf{x}) \sim \text{IID}(0, \sigma^2(\mathbf{x})) \tag{1}$$

where $\mu(\boldsymbol{x})$ is the mean of the stochastic process $Y(\boldsymbol{x})$, and $\delta(\mathbf{x})$ is the additive *noise*, which is assumed independently and identically distributed (IID) with mean zero and variance $\sigma^2(\mathbf{x})$. 'Ordinary' Kriging—to which we limit ourselves—further assumes a *stationary covariance process* for $Y(\mathbf{x})$ in (1); i.e., the expected values $\mu(\mathbf{x})$ are a constant μ and the covariances of $Y(\mathbf{x}+\mathbf{h})$ and $Y(\mathbf{x})$ depend only on the Euclidean distance (lag) $\| \mathbf{h} \| = \| (\mathbf{x}+\mathbf{h}) - (\mathbf{x}) \|$. (The assumption $\mu(\mathbf{x}) = \mu$ is standard in Ordinary Kriging, and does not imply a flat response surface; see Sacks et al. 1989.)

The Kriging *predictor* for the unobserved (non-simulated) input (say) $\mathbf{x}_0$—denoted by $\hat{Y}(\mathbf{x}_0)$— is a weighted linear combination of all the *n* observed outputs:

$$\hat{Y}(\mathbf{x}_0) = \sum_{i=1}^{n} \lambda_i \cdot Y(\mathbf{x}_i) = \boldsymbol{\lambda}' \cdot \mathbf{Y} \tag{2}$$

with $\sum_{i=1}^{n} \lambda_i = 1$, $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_n)'$ and $\mathbf{Y} = (y_1, \ldots, y_n)'$. To select these weights, Kriging derives the Best Linear Unbiased Predictor (BLUP), which (by definition) minimizes the MSE of the predictor:

$$\min_{\lambda} \left\{ \text{MSE}\left(\hat{Y}(\mathbf{x}_0) \right) \right\} = \min_{\lambda} \left\{ E\left(Y(\mathbf{x}_0) - \hat{Y}(\mathbf{x}_0) \right)^2 \right\}. \tag{3}$$

Obviously, this solution depends on the output's covariances. It can be proven that the optimal weights in (2) resulting from (3) are

$$\boldsymbol{\lambda}' = \left(\boldsymbol{\gamma} + \mathbf{1} \frac{1 - \mathbf{1}' \boldsymbol{\Gamma}^{-1} \boldsymbol{\gamma}}{\mathbf{1}' \boldsymbol{\Gamma}^{-1} \mathbf{1}} \right)' \boldsymbol{\Gamma}^{-1} \tag{4}$$

with the following symbols:

$\boldsymbol{\gamma}$ is the vector of covariances between the outputs at the input to be predicted and at the inputs already observed, so $\boldsymbol{\gamma} = (\gamma(\mathbf{x}_0 - \mathbf{x}_1), \ldots, \gamma(\mathbf{x}_0 - \mathbf{x}_n))'$;

$\mathbf{1} = (1, \ldots, 1)'$ is the vector with n ones;

$\boldsymbol{\Gamma}$ is the $n \times n$ matrix whose element (i, j) is the (co)variance at the inputs already observed $\gamma(\mathbf{x}_i - \mathbf{x}_j)$ with $i, j = 1, \ldots, n$.

Note that the weights in (4) vary with $\mathbf{x}_0$ (input to be predicted), whereas regression analysis uses the same estimated metamodel for all inputs $\mathbf{x}$.

Note further that the literature on (deterministic) simulation speaks of covariances and corresponding correlations, whereas the geostatistics literature speaks of the *variogram*, defined as $2\gamma(\mathbf{h}) = var(Y(\mathbf{x} + \mathbf{h}) - Y(\mathbf{x}))$. Since we shall use the Matlab Kriging toolbox DACE—made available free of charge by Lophaven, Nielsen, and Søndergaard (2002)—we avoid the term variogram. (Recent alternative free software is made available via http://www.stat.ohio-state.edu/~comp_exp/; see Santner, Williams, and Notz 2003.)

We emphasize that in practice the covariances $\boldsymbol{\gamma}$ and $\boldsymbol{\Gamma}$ in (4) are unknown so they must be *estimated*. The classical estimator for $\gamma(\mathbf{h})$ is $\hat{\gamma}(\mathbf{h}) = \sum_{N(\mathbf{h})} (Y(\mathbf{x}_i) - Y(\mathbf{x}_j))^2 / (2N(\mathbf{h}))$, where $|N(\mathbf{h})|$ denotes the number of distinct pairs in $N(\mathbf{h}) = \{(\mathbf{x}_i, \mathbf{x}_j) : \mathbf{x}_i - \mathbf{x}_j = \mathbf{h}\}$. Consequently, the weights in (4) become random variables (say) $\hat{\boldsymbol{\lambda}}$. These weights make the Kriging predictor resulting from (2) *non-linear*. This characteristic is often neglected in the Kriging literature. In general, non-linear functions of random variables are hard to analyze—a simple computer-intensive solution is bootstrapping; see Efron and Tibshirani (1993).

Ignoring the randomness of the estimated optimal weights $\hat{\boldsymbol{\lambda}}$ tends to *underestimate* the true variance of the Kriging predictor. For example, in the bivariate normal case this follows from the formula for the conditional variance, namely $\text{var}(Y \mid X) = (1 - \rho^2) \cdot \text{var}(Y)$; see, for example, Kreyszig (1970, p. 343). To tackle this problem, Cressie (1993, p. 146)

proposes *cross-validation*. Cross-validation is also used by Kleijnen and Van Beers (2004) for deterministic simulation. For deterministic simulation, Den Hertog, Kleijnen, and Siem (2005) apply parametric bootstrapping—assuming normally distributed prediction errors—and find that ignoring the randomness of the Kriging weights leads to serious errors. Because random simulation may have non-normal outputs (for example, queueing simulations have distributions with heavy right-hand tails), we use distribution-free bootstrapping—as we shall explain in Section 4.

3. DOE and Kriging

By definition, an experimental *design* is a set of n combinations of k factor values. These combinations are usually bounded by 'box' constraints: $a_j \le x_j \le b_j$ with $a_j, b_j \in R$ and $j = 1, \ldots, k$. The set of all feasible combinations is called the *experimental region* (say) H. We suppose that H is a k-dimensional unit cube, after rescaling the original rectangular area (see Section 1).

Our goal is to find the 'best' design for Kriging predictions within H; the Kriging literature proposed several criteria (see Sacks et al. 1989, p. 414). Most of these criteria are based on the predictor's MSE. Most progress has been made for the IMSE (see Bates et al. 1996):

$$IMSE = \int_H \mathrm{MSE}\left(\hat{Y}(\mathbf{x})\right)\phi(\mathbf{x})d\mathbf{x} \tag{5}$$

where MSE follows from (3), and $\phi(\mathbf{x})$ is a given weight function—usually assumed to be a constant.

To evaluate a design, Sacks et al. (1989, p. 416) compare the predictions with the known output values of a *test set* consisting of (say) N inputs. Assuming a constant $\phi(\mathbf{x})$ in (5), the IMSE can then be estimated by the Empirical IMSE (EIMSE):

$$EIMSE = \frac{1}{N}\sum_{i=1}^{N}\left(\hat{y}_i(\mathbf{x}) - y_i(\mathbf{x})\right)^2 . \tag{6}$$

Besides this EIMSE, we will also study the *maximum* MSE; that is, we also consider risk-averse users (also see Van Groenigen, 2000). So IMSE—defined in (5)—is replaced by

$$MaxMSE = \max_{\mathbf{x} \in H} \left\{ MSE\left(\hat{Y}(\mathbf{x})\right) \right\} \quad (7)$$

and EIMSE in (6) by

$$EMaxIMSE = \max_{i \in \{1, \ldots, m\}} \left\{ \left(\hat{y}_i(\mathbf{x}) - y_i(\mathbf{x}) \right)^2 \right\}. \quad (8)$$

4. Sequential DOE

We devise the following sequential DOE procedure with eight steps, which we illustrate through the M/M/1 model with experimental region $H = \{\rho : 0.1 \leq \rho \leq 0.9\}$ where ρ denotes the traffic rate.

Step 1. We start with a small *pilot design* with (say) n_0 input combinations; for example, n_0 = 5. We select the specific n_0 values such that they are equally spread over the experimental region. There are various 'space filling' designs; for example, LHS designs. In the first example in Section 5—namely the M/M/1—we use a *maximin* design, which (by definition) maximizes the minimum distance between any two points of the design; see Koehler and Owen (1996, p. 288). So in this example, we select the traffic rates $x_i \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$ ($i = 1, \ldots, 5$).

Step 2: For each input value x_i, we initially generate (say) m_0 IID *replicates*—because bootstrapping requires IID observations; see Efron and Tibshirani (1993). To obtain IID observations in our M/M/1 simulation example, we apply *renewal* (regenerative) analysis (see, for example, Kleijnen and Van Groenendaal 1992, and Law and Kelton 2000). As 'the' renewal state, we choose the idle (empty) state. We therefore start the simulation run in the empty state—for each traffic rate x_i. Next we observe m_0 cycles—each with (random) *cycle lengths* (say) L_i (the higher x_i, the higher L_i tends to be). Besides the m_0 cycle lengths $L_{i;j}$ (j = 1, …, m_0) per traffic rate x_i, we observe the sum of the waiting times over that cycle:

$$sw_{i;j} = \sum_{t=1}^{L_{i;j}} w_{i;j;t} \quad (i = 1, \ldots, n_0;\ j = 1, \ldots, m_0). \quad (9)$$

To reduce the variance when comparing the (random) outputs for different inputs (i.e., to improve the signal/noise ratio), we use *common random numbers* (CRN). This is a popular variance reduction technique (VRT). It is well known that—in M/M/1 simulation—the variance decreases substantially if the PRN (say) r_t are manipulated as follows: successive PRN are used alternatively to simulate the arrival time (say) a and the service time s; in other words, $a_t = -\ln r_{2t-1} E(a)$ and $s_t = -\ln r_{2t} E(s)$ (t = 1, 2, …). The correlation coefficients for the average waiting times of two neighboring traffic rates turn out to be very high, namely roughly 0.99.

To generate the PRN, we use the Matlab command 'rand'. To initialize the PRN, we set the Matlab generator (rather arbitrarily) to its initial state $s_0 = 0$. The Matlab web site further states: 'The uniform random number generator in MATLAB 5 (and above) uses a lagged Fibonacci generator, with a cache of 32 floating point numbers, combined with a shift register random integer generator. The integer generator uses shifts and exclusive OR's.'; see (http://www.mathworks.com/support/solutions/data/8542.shtml) and also Moler (1995).

For further details on CRN, VRT, and PRN we refer to Law and Kelton (2000).

Step 3. Based on these m_0 bivariate IID outputs $(L_{i;j}, sw_{i;j})$ (j = 1, …, m_0) per input value x_i, we estimate the mean waiting times through

$$\bar{y}_i(m_0) = \frac{\sum_{j=1}^{m_0} sw_{i;j}}{\sum_{j=1}^{m_0} L_{i;j}} \quad . \tag{10}$$

This *ratio estimator* is consistent; for references see again Kleijnen and Van Groenendaal (1992) and Law and Kelton (2000). We do not try to improve the small-sample performance of this estimator (for example, through jackknifing—which is closely related to bootstrapping), because this estimator suffices for our Kriging metamodel.

To estimate the *precision* of the estimate defined in (10), we use the following probability statement that holds asymptotically per input value x_i:

$$P\left\{\bar{y}_i(m_0) - t_{m_0-1;\,1-\alpha/2} \cdot \frac{\hat{\sigma}_i / \sqrt{m_0}}{\bar{L}_i} \le E(w_i) \le \bar{y}_i(m_0) + t_{m_0-1;\,1-\alpha/2} \cdot \frac{\hat{\sigma}_i / \sqrt{m_0}}{\bar{L}_i}\right\} = 1-\alpha \tag{11}$$

where $\hat{\sigma}_i^2 = \hat{\text{var}}(sw_i) + \bar{y}_i^2 \cdot \hat{\text{var}}(L_i) - 2\bar{y}_i \cdot \hat{\text{cov}}(sw_i, L_i)$ and $\bar{L}_i = \sum_{j=1}^{m_0} L_{i;j} / m_0$; again see Kleijnen and Van Groenendaal (1992). Note that this interval does not have an asymptotic *joint* (or experimentwise) probability (1- α) over all simulated input values.

Next, we add replicates one-at-a-time—*sequential sampling*—until the desired half-width of the interval in (11) has reduced to a prefixed relative error (say) δ ; for example, δ = 0.15 (again see Kleijnen and Van Groenendaal 1992 and Law and Kelton 2000). We denote the final number of replicates per input x_i by m_i . This gives the average output $\bar{y}_i(m_i)$ per input x_i based on m_i replicates; see (10) with m_0 replaced by m_i .

Step 4. Based on these n_0 average outputs $\bar{y}_i(m_i)$ for the n_0 inputs x_i , we compute the *Kriging predictors* for the expected outputs of a new set of (say) n^c *candidate* input values x_g^c (g = 1, …, n^c). We again select these candidates in a *space-filling* way; in the M/M/1 example, we choose the candidate inputs halfway between two old neighboring inputs so we avoid extrapolation: $x_g^c = (x_g + x_{g+1})/2$ (with $g = 1, \ldots, n_0 - 1$).

By definition, the Kriging predictor is a weighted linear combination of all outputs already observed; see (2). So now Kriging weights the n_0 values already observed in steps 1 through 3:

$$\hat{y}(\mathbf{x}_g^c) = \sum_{i=1}^{n_0} \lambda_i \cdot \bar{y}(\mathbf{x}_i) \tag{12}$$

with $\sum_{i=1}^{n_0} \lambda_i$ = 1. To estimate the weights λ_i in (12), Kriging uses the old data set $(x_i, \bar{y}_i(m_i))$ (i = 1, …, n_0) . To estimate the variance of this non-linear predictor, we use bootstrapping—as follows.

Step 5. Per input x_i , we *bootstrap* the m_i bivariate IID outputs ($L_{i;j}$, $sw_{i;j}$); i.e., we resample—with replacement—the outputs resulting from steps 1 through 3. We denote these bootstrap observations by the superscript * (as is traditional in the bootstrap literature):

$$\{(sw_{i;1}^*, L_{i;1}^*), \ldots, (sw_{i;m_i}^*, L_{i;m_i}^*)\} . \tag{13}$$

Using these bootstrapped observations and (10), we compute the bootstrap averages:

$$\bar{y}_i^*(m_i) = \frac{\sum_{j=1}^{m_i} sw_{i;\,j}^*}{\sum_{j=1}^{m_i} L_{i;\,j}^*} \ . \tag{14}$$

Using the bootstrapped I/O data $(x_i, \bar{y}_i^*(m_i))$ $(i = 1, \ldots, n_0)$ and (12), we compute the bootstrapped Kriging predictor:

$$\hat{y}^*(\mathbf{x}_g^c) = \sum_{i=1}^{n_0} \lambda_i^* \cdot \bar{y}^*(\mathbf{x}_i) \, . \tag{15}$$

We again estimate the bootstrap weights λ_i^* in (15) through the Matlab Toolbox DACE; see Section 2.

Note that DACE aims to obtain the maximum likelihood estimator (MLE) of the Kriging weights λ_i^* in (15). For the numerical search that leads to this MLE, DACE uses starting values. As starting values, we use the MLE for λ_i based on the original I/O data in (12).

Step 6. The resampling per input x_i in step 5 is repeated (say) *B* times (this *B* is called the bootstrap sample size). Hence, (13) through (15) give $\hat{y}_b^*(\mathbf{x}_g^c)$ with *b* = 1, …, *B*.

For each of the n^c candidate inputs x_g^c, we compute the bootstrap variance of the Kriging predictor $\hat{y}_g^{c*}$ at x_g^c:

$$\hat{\text{var}}(\hat{y}_g^{c*}) = \frac{1}{B-1} \sum_{b=1}^{B} (\hat{y}_{g;\,b}^{c*} - \bar{\hat{y}}_g^{c*})^2 \tag{16}$$

where $\hat{y}_{g;\,b}^{c*}$ is the predicted value at candidate input x_g^c based on the bootstrapped I/O data $(x_i, \bar{y}_{i;b}^*(m_i))$ $(i = 1, \ldots, n_0)$ and $\bar{\hat{y}}_g^{c*} = \sum_{b=1}^{B} \hat{y}_{g;\,b}^{c*} / B$.

Step 7. We determine which candidate input has the *largest* bootstrap prediction variance (16):

$$v = \arg\left(\max_{g \in \{1, \ldots, n^c\}} \left\{ \hat{\text{var}}(\hat{y}_g^{c*}) \right\} \right), \tag{17}$$

and we add this 'winning' input x_v^c to the old design.

Now, we run the simulation model with this input x_v^c—until we have m_0 replicates for this input. We still apply CRN (so we initialize the PRN with the seed s_0). Furthermore, we again start with the empty system as the renewal state. We continue the simulation until the confidence interval reaches the threshold δ ; see (11).

Step 8. We *repeat* the steps 4 through 7—until we have reached a stopping criterion. In other words, we bootstrap the old I/O set augmented with the candidate selected in step 7. We select a new set of candidates. For these candidates, we compute the Kriging predictors and their bootstrap variances. Alternative stopping criteria may be: (i) the computer budget has been exhausted, (ii) the project has reached its deadline, (iii) the precision of the Kriging metamodel is acceptable.

We observe that adding one point at a time—as we do in our sequential DOE—is not necessarily optimal. However, it is a simple—albeit myopic—heuristic; also see Banjevic and Switzer (2002), who refer to Ferri and Piccioni (1992).

5. Two examples

We test our customized sequential design (CSD) through two classic academic simulation models, namely the M/M/1 model and an (*s*, *S*) model.

5.1. M/M/1 model

An M/M/1 has as true I/O function the hyperbole

$$y = \frac{x}{1-x} \text{ with } 0 < x < 1 \tag{18}$$

where *y* denotes the expected steady-state waiting time assuming a unit service rate, and *x* denotes the traffic rate .

We apply the procedure described in section 4, selecting the following parameters.

Step 1: We select a pilot design of size $n_0 = 5$.

Step 2: We obtain $m_0 = 10$ replicates to get initial estimates of the variances; we select as the initial PRN seed $s_0 = 0$.

Step 3: We experiment with two values for the precision, namely $\delta = 0.05$ and $\delta = 0.15$, and two values for the type-I error rate, namely $\alpha = 0.01$ and 0.05—so (11) gives four confidence intervals. For higher traffic rates (say, $x > 0.7$), the numbers of cycles and the cycle lengths may be very large. To limit computer time, we limit the number of cycles ($L_{i;j}$) to 1000. This limit preserves the renewal property, but may decrease the precision δ.

Step 6: We experiment with the bootstrap sample sizes: $B = 50$ and $B = 100$.

Step 8: We experiment with a stopping criterion that specifies that the total design size is either $n = 10$ or $n = 50$.

Figure 1 displays simulation results for both our design and a LHS design. This figure is based on the confidence intervals in (11) with $\alpha = 0.05$ and $\delta = 0.15$. The bootstrap sample size is only $B = 50$. The stopping criterion is that $n = 10$ traffic rates have been simulated. This figure corresponds with one scenario (labeled 1) of the eight scenarios in our experiment; see Table 1b below. LHS turns out to simulate fewer 'challenging' inputs; i.e., high traffic rates.

Insert Figure 1

To evaluate our procedure, we use a *test set* with $N = 32$ equidistant traffic rates, namely {0.1125, 0.1375, …, 0.8875} (Sacks et al. 1989 also use test sets to evaluate their procedure). We compare the Kriging predictions of the two designs with the 'true' outputs of the test set, computed from (18). (The two designs may contain some members of the test set, but we ignore this phenomenon.) Figure 2 illustrates the 32 predictions for replicate 1 of scenario 1.

Insert Figure 2

To compare the predictions of our design and LHS, we might use the *EIMSE* criterion, defined in (6). However, the final numbers of replicates in the two designs may differ, so we calculate the *corrected EIMSE*, denoted by *e* later on:

$$e = CEIMSE = C \times \frac{1}{n_t} \sum_{i=1}^{n_t} \left(\hat{y}(x_i^t) - y(x_i^t) \right)^2 , \qquad (19)$$

where C is the ratio of the total number of replicates in the LHS design and in our design, n_t is the number of I/O combinations in the test set (so $n_t = 32$), and x_i^t is the i^{th} input of the test set.

We compute this criterion for eight scenarios; i.e., eight combinations of values of the type-I error rate α, the relative error δ, the bootstrap sample size B, and the final design size n. These scenarios are specified through a 2^{k-p} design with $k = 4$ and $p = 1$. This design is expressed in standardized values in Table 1a (see Kleijnen and Van Groenendaal 1992); note that all columns are orthogonal. The original values are displayed in Table 1b.

Insert Tables 1a and 1b

To decrease the randomness of *CEIMSE* in (19), we replicate each scenario in Table 1 $R = 5$ times. To ensure that the PRN streams do not overlap, we start Matlab's PRN generator in the initial state $s_0 = 0$ (using the command RAND('state', 0)) in the first replication of each scenario. Next we save the generator's state of the scenario that requires the largest number of simulation runs; we use that state as the initial state for each of the eight scenarios in the next replication, and so on. Table 2a shows the $R = 5$ CEIMSEs per scenario, denoted by e_r $(r = 1, \ldots, R)$, for the Customized Sequential Design; Table 2b shows e_r for LHS.

Insert Tables 2a and 2b

We analyze the results in Table 2 as follows. Comparing Tables 2a and 2b shows that our designs do not have smaller CEIMSE than LHS designs, in *all* cases (scenarios and replicates). More precisely, our designs give better results only if the design size n is 'small'; see the scenarios 1, 5, 6, and 7. But it is exactly these cases that we are interested in, since (as we stated in Section 1) we focus on 'expensive' simulations, which imply that big design sizes are infeasible. So, we compute the differences

$$d_{i;r} = e_{i;r;\,LHS} - e_{i;r;\,CSD} \text{ with } i = 1, \ldots, 8; \quad r = 1, \ldots, 5. \tag{20}$$

Lumping all scenarios together, the Student *t* test does not give significant differences at a type-I error rate of 5% (the variation of the differences $d_{i;r}$ is large). However, Figure 3 suggests that each of the four scenarios with small *n* (design size) gives significantly positive differences. We therefore investigate which factors explain the performance of our design relative to LHS, as follows.

Insert Figure 3

Remember that we have the $k = 4$ factors corresponding with α, δ, *B*, and *n*. So we estimate the first-order polynomial, which has the main effects β_j:

$$d_{ir} = \beta_0 + \sum_{j=1}^{k} \beta_j x_{ij} + \varepsilon_{ir} \,. \tag{21}$$

We wish to account for variance heterogeneity: $\text{var}(\varepsilon_i) \neq \sigma^2$. Moreover we use CRN, so d_{ir} and $d_{i'r}$ $(i' = 1, \cdots, 8)$ are not independent. Therefore we compute the OLS estimator of the parameters in (21) per replication:

$$\hat{\boldsymbol{\beta}}_r = (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{d}_r \tag{22}$$

where $\boldsymbol{X}$ is the 8×5 matrix following from (21) and Table 1a. This gives the average OLS estimator based on all $R = 5$ replications:

$$\bar{\hat{\boldsymbol{\beta}}} = \frac{1}{R}\sum_{r=1}^{R} \hat{\boldsymbol{\beta}}_r \,. \tag{23}$$

Hence the standard error for the *j*th main effect is

$$s(\bar{\hat{\beta}}_j) = \frac{s(\hat{\beta}_j)}{\sqrt{R}} = \frac{\sqrt{\sum_{r=1}^{R} (\hat{\beta}_{j;r} - \bar{\hat{\beta}}_j)^2 \Big/ (R-1)}}{\sqrt{R}} \,, \tag{24}$$

so the Student statistic with $\nu = R - 1$ degrees of freedom is

$$t_{j;\nu} = \frac{\bar{\hat{\beta}}_j - \beta_j}{s(\bar{\hat{\beta}}_j)} \quad . \tag{25}$$

This statistic assumes normality, which probably holds because the Central Limit Theorem may be applied.

The classic null-hypothesis is that $\beta_j = 0$ (j = 1, …, 4) in (21). We display the corresponding *t*-statistics defined by (25) in Table 3 for three values of the type-I error rate, namely 0.10, 0.05, and 0.01.

Insert Table 3

Table 3 shows that the design size n (factor 4) has a significant negative effect on the difference d (for any of the three type-I error rates); i.e., the advantage of our design becomes smaller as the design size n increases. Further, the bootstrap sample size B (factor 3) has no significant effect: our procedure uses the bootstrap only to estimate which candidate input has the largest variance of the Kriging predictor; see (17). So in practice the smaller size, B = 50, may be used. (Most bootstrap applications require the estimation of the whole distribution function, so B is much higher than 50; for example, B = 1000.) Changes in α and δ (factors 1 and 2) affect the number of replicates, but this effect is incorporated in CEIMSE via the factor C; see (19).

Risk-averse users may be guided by EMaxIMSE, defined in (8). Again, our designs outperform LHS designs for the smaller design sizes n. Table 4a shows the five EMaxIMSE values for scenario i, denoted by $e_{i;r}^{\max}$ for our design, and Table 4b shows the analogous values for LHS; Figure 4 shows the differences, $d_{i;r}^{\max} = e_{i;r;LHS}^{\max} - e_{i;r;CSD}^{\max}$.

Insert Tables 4a and 4b, and Figure 4

Note that m (number of required cycles) indeed increases with x (traffic rate). For example, for the precision requirements α = 0.05 and δ = 0.15, x = 0.1 requires 489 cycles,

whereas x = 0.9 requires the maximum number of cycles, namely 1000; see Figure 5. Moreover, a cycle is likely to be longer as the traffic rate increases. For example, if x = 0.1 then the average cycle length is $\overline{L} = 4.8$ for $m_0 = 10$ replicates; if x = 0.9 then $\overline{L} = 45.9$. For a high traffic rate, the maximum number of cycles (1000) is reached, in this figure. For higher accuracy (δ = 0.05) this maximum is also reached for moderate traffic rates.

Insert Figure 5

A question about our design might be: is the concentration of the simulation runs in the input range with high traffic rates caused by the high signal (E(y)) or the high noise (var(y)) (both the mean and the variance of the M/M/1's steady-state waiting time increase with the traffic rate)? To answer this question, we run some Monte Carlo experiments inspired by the M/M/1 model. In these experiments we use the relative precision δ = 0.15, the type-I error rate α = 0.05, and the final design size n = 15. We use the same PRN seed for the same macro-replicate of the four experiments. We run six macro-replicates; the results across the six macro-replicates look very much alike, so—to save space—we do not display the figures for all macro-replicates; Figure 6 gives results for one macro-replicate.

Insert Figure 6

(a) *Increasing signal and constant noise*: $y = x/(1-x) + r$ with $0.1 \le x \le 0.9$ and $r \in U(-1, 1)$; in other words, the signal follows (18), but the noise is uniformly distributed between –1 and 1, for any input value x. Figure 6(a) shows that our design allocates its runs to the area with rapidly changing signal—as our design did for the M/M/1 in Figure

(b) *Constant signal and increasing noise*: $y = 5 + 10rx$. Figure 6(b) shows that our design again allocates its runs to the high input values with high noise.

(c) *Constant signal and constant noise*: $y = 5 + r$. Figure 6(c) shows that now our design spreads its runs uniformly across the experimental area.

(d) *Increasing signal and decreasing noise*: $y = x/(1-x) + r/(10x)$. Figure 6(d) shows that now our design allocates most of its runs to the middle of the experimental area. Our explanation is that the increasing signal pulls the runs to the high input values, whereas the decreasing noise pulls them to the low values—so that the net result is a 'compromise'.

5.2. (*s*, *S*) inventory model

In an (*s*, *S*) model (with $s < S$) with random demand *D*, the inventory *I* is replenished to the order up-to level *S* whenever the inventory decreases to a value smaller than the reorder level *s*; i.e., the order quantity *Q* is

$$Q = \begin{cases} S - I & \text{if } I < s \\ 0 & \text{if } I \geq s \end{cases}.$$

There are several variations on this basic model, but we simulate Law and Kelton (2000, p. 60, 651)'s example 12.9—which has the following features. Times between demands are IID exponential random variables with a mean of 0.1 month. If a demand arrives, its size is given by the probability function

D	1	2	3	4
$\Pr\{D\}$	$\frac{1}{6}$	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{6}$

The inventory is reviewed at the beginning of each month. Law and Kelton define an auxiliary variable $d = S - s$ to estimate the optimal values for *s* and *S*; the (re)order quantity, however, is not a fixed quantity (the order quantity *Q* varies with the actual 'inventory position', defined as stock on hand, minus customer backorders, plus outstanding supplier orders; see Bashyam and Fu (1998)). The lead-time of an order is uniformly distributed between 0.5 and 1 month. Demand is satisfied immediately if the inventory level *I* is at least as large as the demand size *D*. Otherwise, the demand is—possibly partly—backlogged and delivered as soon as the inventory is replenished. The backlog costs are \$5 per month per item backlogged. Holding costs per item per month are \$1. Ordering costs consist of a setup cost of \$32 per order plus incremental costs of \$3 per item.

Law and Kelton simulate the system for 120 months, starting with an initial inventory $I(0) = 60$; i.e., this simulation model is *terminating* (example 1 estimates a steady-state mean of an M/M/1). Law and Kelton obtain five replicates for each of the 36 combinations formed by $s = 0, 20, 40, 60, 80, 100$ and $d = 0, 20, 40, 60, 80, 100$. Based on these 180 I/O data, they fit the following *second-order polynomial* regression (meta) model for the average monthly total costs called *R*:

$$\hat{R}(s, d) = 188.51 - 1.49s - 1.24d + 0.014sd + 0.007s^2 + 0.010d^2 . \quad (26)$$

They compare this model's predictions with the 'true' $E(R)$ estimated from 10 replicates for each of 420 new and old combinations formed by $s = 0, 5, 10, \ldots, 100$, and $d = 5, 10, 15, \ldots, 100$.

We, however, replace (26) by a Kriging model, fitted to the same I/O data (implying 36 average outputs), and compare our Kriging predictions with the 'true' outputs. We find that our Kriging model gives more accurate predictions than the regression model (26); see the Appendix for details.

Next, we change the design from Law and Kelton's *grid* (with 16 combinations of the two inputs *s* and *d* with $(s, d) \in [20, 80] \times [20, 80]$) into our design (with the same final design size, namely 16); see Figure 7.

Insert Figure 7

Like Law and Kelton, we obtain 5 replications per input combination. Next, we fit a Kriging model, and predict 81 'true' outcomes for the test set $(s, d) \in \{10, 20, \ldots, 90\} \times \{10, 20, \ldots, 90\}$ (a subset of Law and Kelton's 'true' set). Again, we calculate EIMSE and EMaxIMSE defined in (6) and (8). To reduce noise, we repeat this procedure 5 times (using non-overlapping PRN streams) for our designs and LHS. Our designs give substantial better EIMSE and EMaxIMSE; see Table 5.

Insert Table 5

We conclude that in this example, our sequential design also gives more accurate Kriging predictions than LHS with a fixed design size.

6. Conclusions and future research

In practice, simulation often requires much computer time per run (or replicate)—so it is desirable to have an efficient experimental design for interpolation. It is well known in mathematical statistics that sequential designs are more efficient than fixed-sample designs. Our specific sequential designs add as the next input to be simulated, the input with the

maximum estimated variance for the output predicted at specific candidate inputs. To obtain such predictions, we use Kriging; to estimate the variances of the Kriging predictors, we use bootstrapping. We applied this procedure to estimate (i) the expected steady-state waiting time in M/M/1 simulation, and (ii) the expected cost in terminating inventory (*s*, *S*) simulation. We compared the Kriging prediction errors of our sequential designs and those of fixed-sample LHS. Our results show that our procedure gives indeed smaller prediction errors.

In future research, (asymptotic) proofs of the performance of our procedure might be derived. More experimentation and analyses may be done to derive rules of thumb for our procedure's parameters, such as the initial design size n_0 and the initial number of replicates m_0. Our procedure may be applied to examples more complicated than the M//M/1 queueing model or the (*s*, *S*) inventory model. Stopping rules based on a measure of accuracy or precision may be investigated. Besides LHS, other designs with prefixed sizes may be explored; for example, min-max designs. Besides Ordinary Kriging, other metamodels may be used to analyze the I/O data. For example, the 'optimal' weights in Ordinary Kriging assume that the predictors equal the average outputs at the inputs already observed; dropping this constraint implies that new Kriging software must be developed. New Kriging weights may be derived, replacing the IMSE criterion by the maximum squared error criterion. Besides Kriging, other interpolation models may be used; for example, linear or nonlinear regression metamodels. We focus on sensitivity analysis; searching for the optimal input of the simulation model requires further research.

Appendix

Law & Kelton's (2000, p. 651) data set consists of 5 replicates for each of the 16 input combinations formed by $s_i \in \{20, 40, 60, 80\}$ and $d_j \in \{20, 40, 60, 80\}$ (this set is a subset of the one in the main text). Based on this input set, we find the following estimates

$$\hat{\boldsymbol{\beta}} = (130.6285, -0.2630, -0.5303, 0.0088, 0.0052, 0.0038)',$$

which agrees with their values up to two decimals.

As a test set (used to compare regression and Kriging metamodels), we use their 'true' I/O set, which consists of 10 replicates of each of 420 = 21×20 input combinations with

$s_i \in \{0, 5, 10, \ldots, 100\}$ and $d_j \in \{5, 10, 15, \ldots, 100\}$. For the regression model we find an EIMSE of 1450.5, whereas for the Kriging model we find an EIMSE of 1200.7. So Kriging does result in a smaller EIMSE. This EIMSE, however, is still rather large, because we have to extrapolate the data outside the region $[20,80]\times[20,80]$. In general, we strongly recommend avoiding extrapolation when fitting a metamodel; indeed, in simulation it is easy to avoid extrapolation because we can select our own input combinations.

Law and Kelton also use a data set consisting of 180 I/O combinations, namely 5 replicates for each of 36 input combinations with $s_i \in \{0, 20, 40, 60, 80, 100\}$ and $d_j \in \{0, 20, 40, 60, 80, 100\}$. We use their computer program (imported from their web page http://www.mhhe.com/engcs/industrial/lawkelton/student/code.mhtml) to generate the output. Again, we fit both a second-order regression model and a Kriging model. We compare the two fitted models via the 'true' data set. For the regression model, we find an EIMSE of 152.0, whereas for the Kriging model we find an EIMSE of only 14.0 (in this case extrapolation is indeed avoided.

Acknowledgements

We thank Dick den Hertog (Tilburg University) for his comments on an earlier version, which lead to the additional Monte Carlo experiments reported in Section 5, and Ruud Brekelmans (Tilburg University) for helping us to import Law and Kelton's C-program codes into our Matlab program.

References

Banjevic, M. and P. Switzer (2002), Bayesian network designs for variance as a function of the location. *Proceedings of the 2002 JSM Conference, Section on Statistics and the Environment*, New York, NY

Bashyam, S. and M.C. Fu (1998), Optimization of (s, S) inventory systems with random lead times and a service level constraint. *Management Science*. 44, no. 12, pp. 243-256

Bates, R.A., R.J. Buck, E. Riccomagno and H.P. Wynn (1996), Experimental design and observation for large systems. *Royal Statistical Society*. 58, no. 1, pp. 77-94

Box, G.E.P., W.G. Hunter and J.S. Hunter (1978), *Statistics for experimenters: an introduction to design, data analysis and model building.* John Wiley & Sons, Inc., New York

Crary, S.B. (2002), Design of computer experiments for metamodel generation, *Analog Integrated Circuits and Signal Processing*, 32, pp. 7-16

Cressie, N.A.C. (1993), *Statistics for spatial data*. John Wiley & Sons, Inc., New York

Efron, B. and R.J. Tibshirani (1993). *An introduction to the bootstrap*. Chapman & Hall, New York

Ferri, M. and M. Piccioni (1992), Optimal selection of statistical units. *Computational Statistics & Data Analysis*, 13, pp. 47-61

Ghosh, B.K. and P.K. Sen (editors), 1991, *Handbook of sequential analysis*. Marcel Dekker, Inc., New York

Den Hertog, D., J.P.C. Kleijnen, and A.Y.D. Siem (2005), The correct Kriging variance estimated by bootstrapping. *Journal of the Operational Research Society* (accepted; preprint: http://center.kub.nl/staff/kleijnen/papers.html)

Journel, A.G. and C.J. Huijbregts (1978), *Mining geostatistics*, Academic Press, London

Kleijnen, J.P.C. (1987), *Statistical tools for simulation practitioners*. Marcel Dekker, Inc., New York

Kleijnen, J.P.C. and D. Deflandre (2005), Validation of regression metamodels in simulation: Bootstrap approach. *European Journal of Operational Research* (in press)

Kleijnen, J.P.C., S.M. Sanchez, T.W. Lucas and T.M. Cioppa (2005), A user's guide to the brave new world of designing simulation experiments. *INFORMS Journal on Computing* (accepted as State-of-the-Art Review)

Kleijnen, J.P.C. and W.C.M. van Beers (2004), Application-driven sequential designs for simulation experiments: Kriging metamodeling. *Journal of the Operational Research Society*, no. 55, pp. 876-883

Kleijnen, J.P.C. and W. van Groenendaal (1992), *Simulation: a statistical* perspective. John Wiley, Chichester (England)

Koehler, J.R. and A.B. Owen (1996), Computer experiments. *Handbook of statistics*, by S. Ghosh and C.R. Rao, vol. 13, pp. 261-308

Kreyszig, E. (1970), *Introductory mathematical statistics: principles and methods.* John Wiley & Sons, Inc., New York

Law, A.M. and W.D. Kelton (2000), *Simulation modeling and analysis, third edition*, McGraw-Hill, Boston

Lophaven, S.N., H.B. Nielsen and J. Søndergaard (2002), A Matlab Kriging toolbox. *Technical report IMM-TR-2002-12*, Technical University of Denmark

McKay, M.D., R.J. Beckman and W.J. Conover (1979), A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 21, no. 2, pp. 239-245 (reprinted in 2000: Technometrics, 42, no. 1, pp. 55-61

Moler, C. (1995), Random thoughts. *MATLAB News & Notes*, pp. 12-13

Myers, R.H. and D.C. Montgomery (2002). *Response surface methodology: process and product optimization using designed experiments; second edition.* Wiley, New York

Park, S., J.W. Fowler, G.T. Mackulak, J.B. Keats, and W.M. Carlyle (2002), D-optimal sequential experiments for generating a simulation-based cycle time-throughput curve. *Operations Research*, 50, no. 6, pp. 981-990

Sacks, J., W.J. Welch, T.J. Mitchell and H.P. Wynn (1989), Design and analysis of computer experiments. *Statistical Science*, 4, no. 4, pp. 409-435

Santner, T.J., B.J. Williams, and W.I. Notz (2003), *The design and analysis of computer experiments*. Springer-Verlag, New York

Sasena, M.J, P. Papalambros, and P. Goovaerts (2002), Exploration of metamodeling sampling criteria for constrained global optimization. *Engineering Optimization* 34, no.3, pp. 263-278

Simpson, T.W., T.M. Mauery, J.J. Korte, and F. Mistree (2001), Kriging metamodels for global approximation in simulation-based multidisciplinary design optimization. *AIAA Journal*, 39, no. 12, 2001, pp. 2233-2241

Van Beers, W. and J.P.C. Kleijnen (2003), Kriging for interpolation in random simulation. *Journal of the Operational Research Society*, no. 54, pp. 255-262

Van Groenigen, J.W. (2000), The influence of variogram parameters on optimal sampling schemes for mapping by Kriging. *Geoderma*, no. 97, pp. 223-236

Williams, B.J., T.J. Santner, and W.I. Notz (2000), Sequential design of computer experiments to minimize integrated response functions, *Statistica Sinica*, 10, 1133-1152

Williams, B.J., T.J. Santner, and W.I. Notz (2002), Sequential design of computer experiments for constrained optimization of integrated response functions, Working Paper. Ohio State University

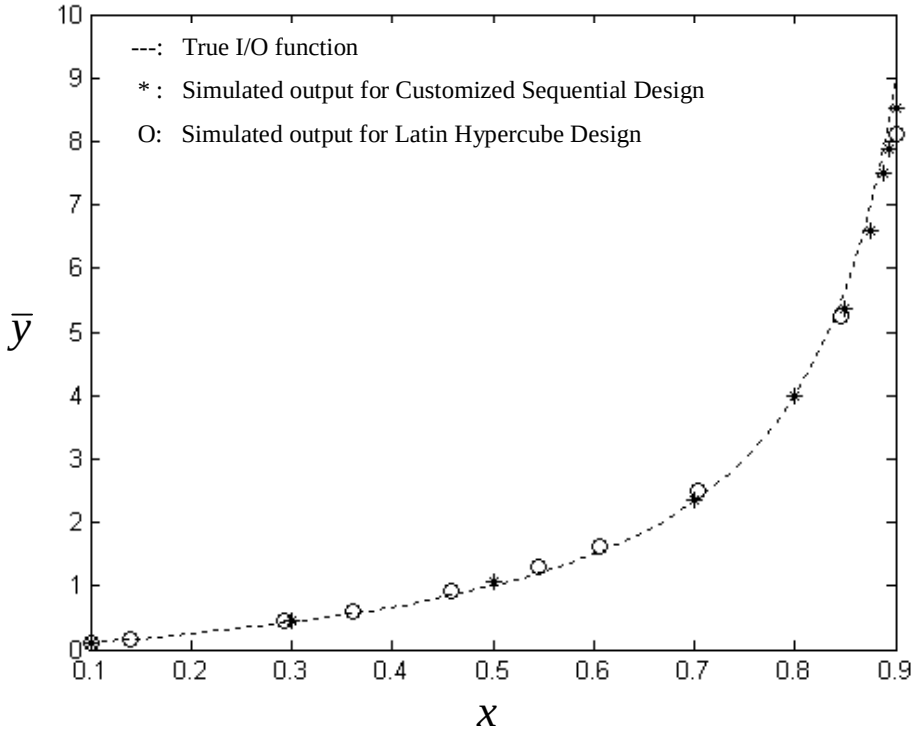

Figure 1: Two designs for M/M/1 with 10 traffic rates x and average simulation outputs $\bar{y}$

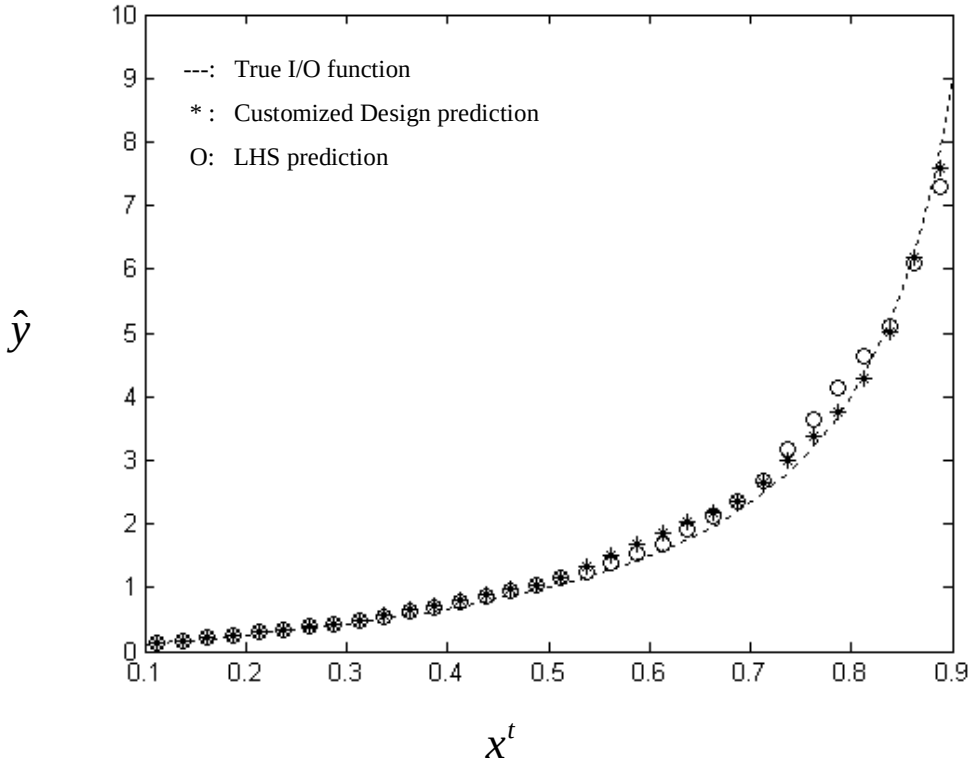

Figure 2: Predictions $\hat{y}$ for the test set for M/M/1, for two designs in replicate 1 of scenario 1

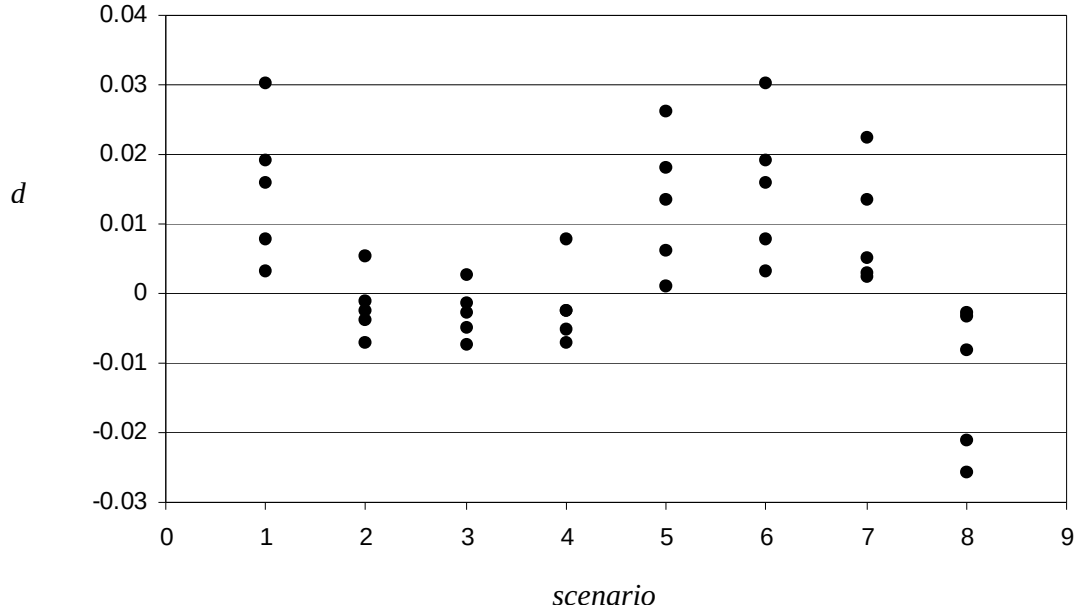

Figure 3: Differences $d_{i;r} = e_{i;r;\,LHS} - e_{i;r;\,CSD}$ for scenario i = 1, …, 8 and replicate r = 1, …, 5

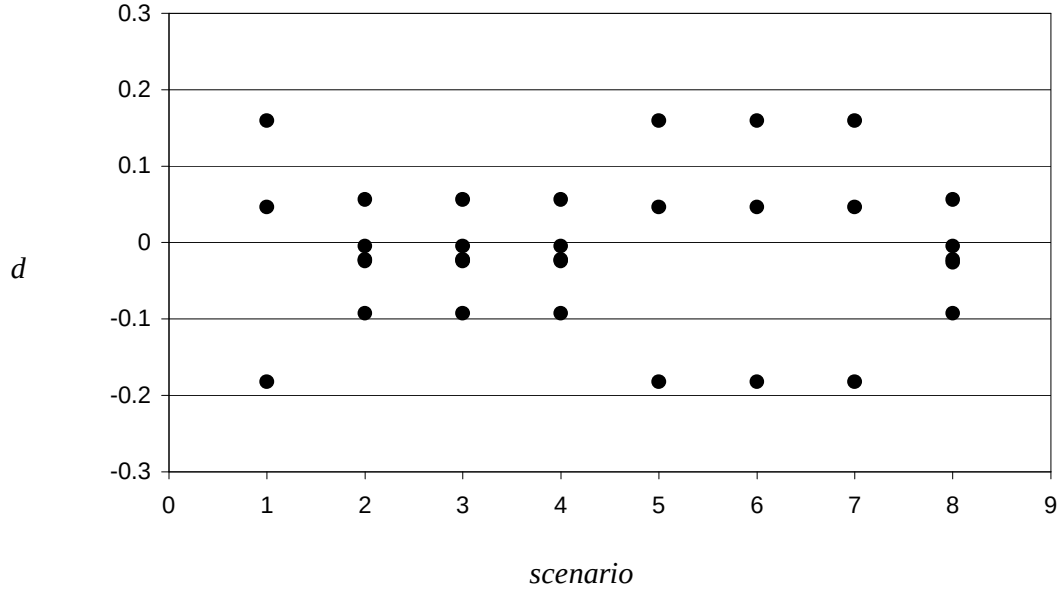

Figure 4: Differences $d_{i;r}^{\max} = e_{i;r;\,LHS}^{\max} - e_{i;r;\,CSD}^{\max}$ for scenario i = 1, …, 8 and replicate r = 1, …, 5

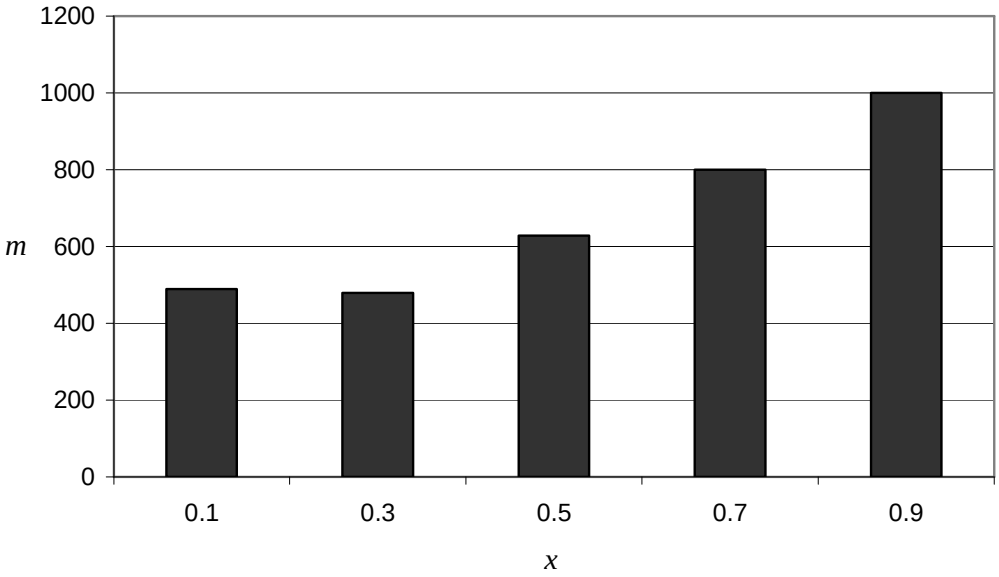

Figure 5: Number of cycles *m* per traffic rate *x* for M/M/1, given $\alpha = 0.05$ and $\delta = 0.15$

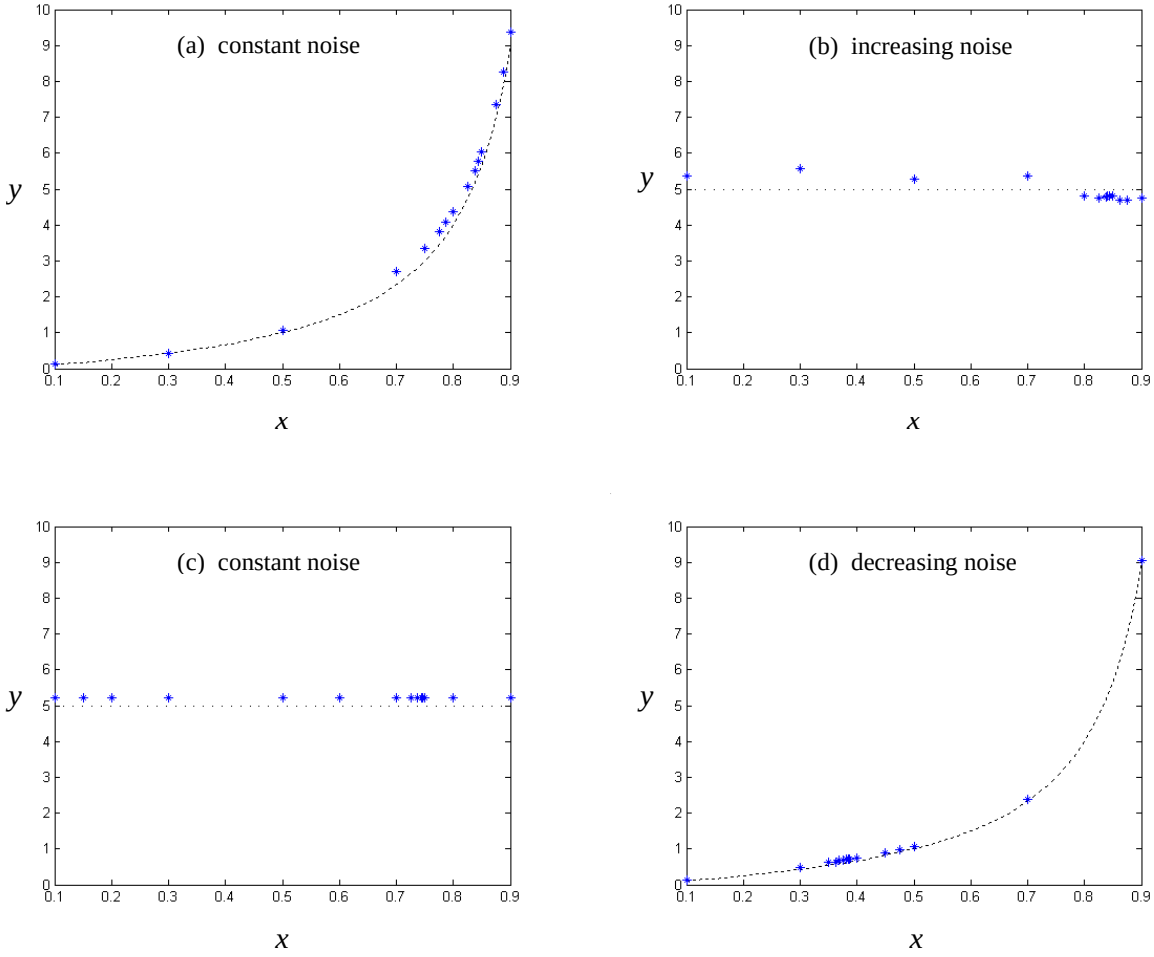

Figure 6: Monte Carlo experiments with four combinations of signal and noise functions;
--- denotes signal and *** denotes I/O of Customized Sequential Design

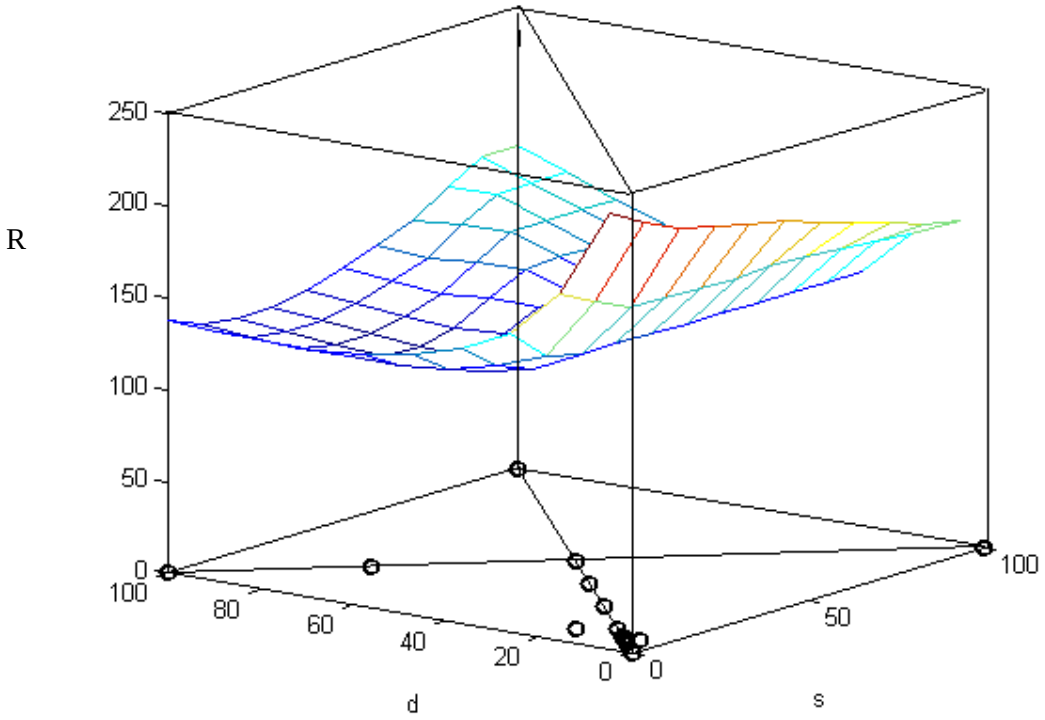

Figure 7: I/O simulation data for (*s*, *S*) inventory model with 16 scenarios denoted by O

Table 1a: A 2^{4-1} design expressed in standardized factor values

factor	α	δ	B	n
scenario	1	2	3	$4 = 1 \cdot 2 \cdot 3$
1	-	-	-	-
2	-	-	+	+
3	-	+	-	+
4	+	-	-	+
5	-	+	+	-
6	+	-	+	-
7	+	+	-	-
8	+	+	+	+

Table 1b: Eight scenarios or combinations of type-I error rate α, relative error δ, bootstrap sample size *B*, and final design size *n*

scenario	α	δ	B	n
1	0.01	0.05	50	10
2	0.01	0.05	100	50
3	0.01	0.15	50	50
4	0.05	0.05	50	50
5	0.01	0.15	100	10
6	0.05	0.05	100	10
7	0.05	0.15	50	10
8	0.05	0.15	100	50

Table 2a: CEIMSE e_r for Customized Sequential Designs
in 8 scenarios replicated 5 times, computed from test set with 32 values

scenario	e_1	e_2	e_3	e_4	e_5
1	0.015026	0.028725	0.005305	0.15052	0.11056
2	0.010669	0.027213	0.010518	0.17480	0.12000
3	0.011028	0.027209	0.010513	0.17480	0.11951
4	0.010669	0.028481	0.010518	0.17636	0.11762
5	0.014915	0.029568	0.005417	0.15051	0.11044
6	0.015026	0.028725	0.005305	0.15052	0.11056
7	0.014645	0.028676	0.004749	0.12363	0.10993
8	0.011019	0.027314	0.010347	0.17486	0.12167

Table 2b: CEIMSE e_r for LHS designs
in 8 scenarios replicated 5 times, computed from test set with 32 values

scenario	e_1	e_2	e_3	e_4	e_5
1	0.045243	0.036466	0.024428	0.15382	0.126451
2	0.003626	0.026059	0.008152	0.17114	0.12551
3	0.003649	0.025891	0.007919	0.17000	0.12209
4	0.003626	0.026059	0.008152	0.17114	0.12551
5	0.041051	0.035814	0.023546	0.15164	0.124033
6	0.045243	0.036466	0.024428	0.15382	0.126451
7	0.037169	0.033886	0.018249	0.12648	0.112403
8	0.002993	0.024671	0.007233	0.14924	0.10047

Table 3: Significance of estimated main effects $\hat{\beta}_j$

	t-statistic	two-sided significance level		
	$t_{j;\nu}$	$\alpha = 0.10$	$\alpha = 0.05$	$\alpha = 0.01$
β_1	-2.2962201	significant	significant	not signif.
β_2	-2.4742393	significant	significant	not signif.
β_3	-1.079914	not signif.	not signif.	not signif.
β_4	-3.8774691	significant	significant	significant

Table 4a: EMaxIMSE $e_r^{\max}$ for Customized Sequential Designs
in 8 scenarios replicated 5 times, computed from test set with 32 values

EMaxIMSE e_i **for Customize Sequential Designs**					
scenario	e_1	e_2	e_3	e_4	e_5
1	0.068502	0.52477	0.024872	0.22247	1.0878
2	0.047377	0.52477	0.1374	0.22247	1.2755
3	0.047378	0.52477	0.1374	0.22247	1.2755
4	0.047377	0.52477	0.1374	0.22247	1.2755
5	0.068502	0.52477	0.024872	0.22247	1.0878
6	0.068502	0.52477	0.024872	0.22247	1.0878
7	0.068502	0.52477	0.024872	0.22247	1.0878
8	0.049059	0.52477	0.1374	0.22247	1.2755

Table 4b: EMaxIMSE $e_r^{\max}$ for LHS designs
in 8 scenarios replicated 5 times, computed from test set with 32 values

EMaxIMSE e_i **for LHS**					
scenario	e_1	e_2	e_3	e_4	e_5
1	0.57114	0.34262	0.1845	0.2689	1.80351
2	0.023245	0.52006	0.11484	0.27878	1.183
3	0.023245	0.52006	0.11484	0.27878	1.183
4	0.023245	0.52006	0.11484	0.27878	1.183
5	0.57114	0.34262	0.1845	0.2689	1.80351
6	0.57114	0.34262	0.1845	0.2689	1.80351
7	0.57114	0.34262	0.1845	0.2689	1.80351
8	0.023245	0.52006	0.11484	0.27878	1.183

Table 5: EIMSE and EMaxIMSE for CSD and LHS for (*s*, *S*) inventory simulation, based on test set with 81 true values

	CSD		**LHS**	
replicate	EIMSE	EMaxIMSE	EIMSE	EMaxIMSE
1	234.2	1724.4	432.9	4282.6
2	319.3	2536.9	686.9	6293.1
3	262.2	1933.3	726.4	6031.1
4	236.2	1732.9	554.5	5017.1
5	213.2	1546.5	666.5	5909.8